IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Daniel J. Magenheimer	§	Conf. No.:	2613
Serial No.:	10/784,065	§	Examiner:	Abdullah Al Kawsar
Filed:	February 20, 2004	§	Art Unit:	2195
For:	Flexible Operating System Operable as Either Native or as Virtualized	§	Atty. Dkt. No.:	200315952-1 (HPC.0500US)

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF PURSUANT TO 37 C.F.R § 41.37

Sir:

The final rejection of claims 1, 3-6, 8, 9, 11-20, 22, 24, 25, 27, 29-31, 33, 35-37, 39, 40, and 42-47 is hereby appealed.

I. REAL PARTY IN INTEREST

The real party in interest is the Hewlett-Packard Development Company, LP. The Hewlett-Packard Development Company, LP, a limited partnership established under the laws of the State of Texas and having a principal place of business at 11445 Compaq Center Drive West, Houston, TX 77070, U.S.A. (hereinafter "HPDC"). HPDC is a Texas limited partnership and is a wholly-owned affiliate of Hewlett-Packard Company, a Delaware Corporation, headquartered in Palo Alto, CA. The general or managing partner of HPDC is HPQ Holdings, LLC.

II. RELATED APPEALS AND INTERFERENCES

None.

III. STATUS OF THE CLAIMS

Claims 1, 3-6, 8, 9, 11-20, 22, 24, 25, 27, 29-31, 33, 35-37, 39, 40, and 42-47 have been finally rejected and are the subject of this appeal.

Claims 2, 7, 10, 21, 23, 26, 28, 32, 34, 38, and 41 have been cancelled.

Claims 5 and 6 have been allowed.

IV. STATUS OF AMENDMENTS

An Amendment under 37 C.F.R. § 1.116 was submitted on July 20, 2010 to fix a typographical error. Entry of this Amendment is appropriate since the scope of the claim has not changed.

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

The following provides a concise explanation of the subject matter defined in each of the independent claims involved in the appeal, referring to the specification by page and line number and to the drawings by reference characters, as required by 37 C.F.R. § 41.37(c)(1)(v). Each element of the claims is identified by a corresponding reference to the specification and drawings where applicable. Note that the citation to passages in the specification and drawings for each claim element does not imply that the limitations from the specification and drawings should be read into the corresponding claim element. Note also that the cited passages are provided as examples, as other passages in the specification or drawings not cited may also be relevant to the corresponding claim elements.

Independent claim 1 recites a computer system comprising:

at least one processor (Fig. 7:701; Spec., p. 21, ¶ [0067], ln. 1-8); and

a flexible operating system executable on the at least one processor to (Spec., p. 4, ¶ [0016], ln. 1-11; p. 20, ¶ [0066], ln. 1-9):

determine whether said flexible operating system is being used as a native operating system or as a virtualized operating system on said computer system, wherein the determining is based on checking a variable set during a boot process of the computer system (Spec., p. 4, ¶ [0016], ln. 3-11); and

execute in a first manner as a native operating system on the computer system in response to detecting that said flexible operating system is being used as the native operation system, and execute in a second manner as a virtualized operating system on said computer system in response to detecting that said flexible operating system is being used as the virtualized operating system (Spec., p. 5, ¶ [0017], ln. 1-13; p. 5, ¶ [0019], ln. 1-16; p. 6, ¶ [0021], ln. 1 - p. 8, ¶ [0025], ln. 18),

wherein said flexible operating system is configured to operate in a non virtualized environment when said flexible operating system is being used as the native operating system, and is configured to operate in a virtualized environment when said flexible operating system is being used as the virtualized operating system (Spec., p. 5, ¶ [0017], ln. 1-13; p. 5, ¶ [0019], ln. 1-16; p. 6, ¶ [0021], ln. 1 - p. 8, ¶ [0025], ln. 18).

Independent claim 11 recites a method comprising:

implementing at least one operating system on a computer system (Fig. 7:500; Spec., p. 4, ¶ [0016], ln. 1-11; p. 20, ¶ [0066], ln. 1-9);

determining, by said computer system, whether said at least one operating system is a native operating system or a guest operating system on a virtual machine, wherein the determining is based on checking a variable set during a boot process of the computer system (Spec., p. 4, ¶ [0016], ln. 3-11);

said at least one operating system operating in a first manner if determined that it is a native operating system, wherein the native operating system operates in a non virtualized environment (Spec., p. 5, ¶ [0017], ln. 1-13; p. 5, ¶ [0019], ln. 1-16; p. 6, ¶ [0021], ln. 1 - p. 8, ¶ [0025], ln. 18); and

said at least one operating system operating in a second manner if determined that it is a guest operating system on a virtual machine, wherein the guest operating system operates in a virtual environment provided by the virtual machine (Spec., p. 5, ¶ [0017], ln. 1-13; p. 5, ¶ [0019], ln. 1-16; p. 6, ¶ [0021], ln. 1 - p. 8, ¶ [0025], ln. 18).

Independent claim 17 recites a computer system comprising:

at least one processor (Fig. 7:701; Spec., p. 21, ¶ [0067], ln. 1-8);

a virtual machine monitor (VMM) (Fig. 1:12; Spec., p. 5, ¶ [0018], ln. 1-13); and

an operating system executable on the at least one processor to (Spec., p. 4, ¶ [0016], ln. 1-11; p. 20, ¶ [0066], ln. 1-9):

determine whether said operating system is running as a virtualized operating system or a native operating system, wherein the determining is based on checking a variable set during a boot process of the computer system (Spec., p. 4, ¶ [0016], ln. 3-11); and

adapt operation of said operating system depending on whether it is running as the virtualized operating system or native operating system, wherein the native operating system is configured to manage hardware resources in a non virtualized environment without the VMM, and wherein the virtualized operating system is configured to manage hardware resources using the VMM (Spec., p. 5, ¶ [0017], ln. 1-13; p. 5, ¶ [0019], ln. 1-16; p. 6, ¶ [0021], ln. 1 - p. 8, ¶ [0025], ln. 18).

Independent claim 27 recites a system comprising:

hardware resources (Fig. 1:13; Spec., p. 5, ¶ [0017], ln. 1-13);

a virtual machine monitor (VMM) (Fig. 1:12; Spec., p. 5, ¶ [0018], ln. 1-13); and

a flexible operating system for managing said hardware resources, wherein said flexible operating system is operable to determine whether it is running in a virtualized environment or in a native, non virtualized environment, wherein the determining is based on checking a variable set during a boot process of the system, wherein said flexible operating system is operable to selectively execute in a first manner if determined that said flexible operating system is running in the native environment and in a second manner if determined that said flexible operating system is running in the virtualized environment, wherein in the first manner said flexible operating system is configured to manage said hardware resources without using the VMM, and wherein in the second manner said flexible operating system is configured to manage said hardware resources using the VMM (Spec., p. 4, ¶ [0016], ln. 1-11; p. 5, ¶ [0017], ln. 1-13; p. 5, ¶ [0019], ln. 1-16; p. 6, ¶ [0021], ln. 1 - p. 8, ¶ [0025], ln. 18).

Claims 35, 37, and 47, set forth below, includes means-plus-function elements, which are identified as required by 37 C.F.R. § 41.37. For each means-plus-function element, the structure, material, or acts described in the Specification as corresponding to each claimed function is set forth by reference to page and line number, and to the drawings, by reference characters.

Independent claim 35 recites a system comprising:

at least one processor (Fig. 7:701; Spec., p. 21, ¶ [0067], ln. 1-8);

a flexible operating system (Spec., p. 4, ¶ [0016], ln. 1-11; p. 20, ¶ [0066], ln. 1-9) executable on the at least one processor and that is capable of acting as either a native operating system or as a virtualized operating system; and

means (Fig. 1:OS) for determining whether the flexible operating system is to be used as a native operating system in a non virtualized environment without a Virtual Machine Monitor (VMM) or as a virtualized operating system in a virtualized environment with the VMM, wherein the determining means checks a variable set during a boot process of the system for determining whether the flexible operating system is being used as the native or as the virtualized operating system (Spec., p. 4, ¶ [0016], ln. 3-11; p. 5, ¶ [0017], ln. 1-13; p. 5, ¶ [0019], ln. 1-16; p. 6, ¶ [0021], ln. 1 - p. 8, ¶ [0025], ln. 18).

Claim 37 recites the system of claim 35 further comprising:

means (Fig. 1:12) for virtualizing resources of said system and multiplexing said resources among one or more virtualized operating systems (Spec., p. 5, ¶ [0018], ln. 1-13).

Claim 47 recites the system of claim 37, further comprising at least one processor (Fig. 1:7:701) configured to selectively set the variable to one of plural values during the boot process, a first of the plural values to indicate that the flexible operating system is to be run in the native environment, and a second of the plural values to indicate that the flexible operating system is to be fun in the virtualized environment (Spec., p. 6, ¶ [0021], ln. 1-6),

wherein the flexible operating system is operable to check the variable during runtime after the boot process (Spec., p. 7, ¶ [0024], ln. 1-3).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A. Claims 1, 11-15, 17, 18, 24, 25, 27, 29-30, 35, 36, 40, and 42-47 were rejected under 35 U.S.C. § 103(a) as unpatentable over Silvester (U.S. Patent No. 7,010,634) in view of Kishi (U.S. Patent No. 5,023,771) and in view of Fish (U.S. Patent No. 6,199,159).

B. Claims 3, 4, 8, 16, 19, 20, 22, 31, 33, 37, and 39 were rejected under 35 U.S.C. § 103(a) as unpatentable over Silvester in view of Kishi and in view Fish, and in view of Bennett (U.S. Patent Publication No. 2004/011732).

C. Claim 9 was rejected under 35 U.S.C. § 103(a) as unpatentable over Silvester in view of Kishi and in view of Fish, and in view of Bennett and further in view of Waldspurger (U.S. Patent No. 6,725,289).

VII. ARGUMENT

The claims do not stand or fall together. Instead, Appellant presents separate arguments for various independent and dependent claims. Each of these arguments is separately argued below and presented with separate headings and sub-headings as required by 37 C.F.R. § 41.37(c)(1)(vii).

A. Claims 1, 11-15, 17, 18, 24, 25, 27, 29-30, 35, 36, 40, and 42-47 were rejected under 35 U.S.C. § 103(a) as unpatentable over Silvester (U.S. Patent No. 7,010,634) in view of Kishi (U.S. Patent No. 5,023,771) and in view of Fish (U.S. Patent No. 6,199,159).0

1. Claims 1, 27, 29, 30, 43.

Claim 1 recites a computer comprising, *inter alia*:

a flexible operating system executable on the at least one processor to:

> determine whether said flexible operating system is being used as a native operating system or as a virtualized operating system on said computer system, wherein the determining is based on checking a variable set during a boot process of the computer system; and
>
> execute in a first manner as a native operating system on the computer system in response to detecting that said flexible operating system is being used as the native operation system, and execute in a second manner as a virtualized operating system on said computer system in response to detecting that said flexible operating system is being used as the virtualized operating system,

> wherein said flexible operating system is configured to operate in a non virtualized environment when said flexible operating system is being used as the native operating system, and is configured to operate in a virtualized environment when said flexible operating system is being used as the virtualized operating system.

Importantly, note that the "flexible operating system" of claim 1 is executable in a first manner under a first condition and in a second manner under a second condition, where the flexible operating system is executable in the first manner as a native operating system, and is executable in the second manner as a virtualized operating system.

None of the references cited by the Examiner provide any hint of such a flexible operating system that executes as a native operating system under a first condition, and executes as a virtualized operating system under a second condition. Specifically, claim 1 recites that the flexible operating system executes in the first manner as the native operating system on the computer system in response to detecting that the flexible operating system is being used as the native operating system, and executes in a second manner as a virtualized operating system on the computer system in response to detecting that the flexible operating system is being used as the virtualized operating system. The determination of whether the flexible operating system is being used as the native operating system or as the virtualized operating system is based on checking a variable set during a boot process of the computer system.

Each of the references (Silvester, Kishi, and Fish) cited by the Examiner refers to **multiple separate** operating systems. Silvester refers to a notebook computer 100 that has a full operating system, and a core computer 101 that has a mini operating system. Silvester, 4:29-37. According to Silvester, when the core computer 101 is undocked, the mini operating system is booted by the core computer. *Id.* On the other hand, when the core computer 101 is docked to the notebook computer 100, the full operating system of the notebook computer is loaded. *Id.* Loading one of two **different and separate** operating systems located in two corresponding

different devices (core computer 101 and notebook computer 100, respectively), has nothing to do with a flexible operating system that can execute as both a native operating system and a virtualized operating system under different respective conditions, as claimed.

Similarly, Kishi also discloses different operating systems. Specifically, Kishi discloses a host operating system 11 as well as separate guest operating systems 16, 17, and 19. The host operating system 11 of Kishi cannot be considered a flexible operating system that is configurable to be used as both a native operating system and a virtualized operating system under different respective conditions. The guest operating systems of Kishi also cannot be considered a flexible operating system as claimed. As specifically taught by Kishi, the host operating system is run on a real CPU. Kishi, 1:20-22. In contrast, guest operating systems are controlled by a virtual machine control program that is run in compliance with the host operating system as a specific job of the host operating system. *Id.*, 1:24-30. As also taught by Kishi, when the virtual machine system is in a non-virtual machine mode, the host operating system 11 is run. *Id.*, 3:51-57. **Alternatively**, when the virtual machine system is operable in the virtual machine operating system mode, the guest operating systems are run. *Id.*, 3:57-63. Thus, according to Kishi, **different** operating systems are run in different modes, in stark contrast to the subject matter of claim 1, where the flexible operating system executes as a native operating system in response to detecting a first condition, and executes as a virtualized operating system in response to detecting a second condition.

The third reference, Fish, cited by the Examiner also discloses two **separate** operating systems, namely a virtual mode operating system 12 and a real mode operating system 10. During the bootup process, a check is made of whether the virtual mode operating system 12 was selected. Fish, 3:2-5. If not, then another operating system is loaded. *Id.*, 3:5-7. Thus, it is clear

that according to Fish there are two separate operating systems that are selectively loaded. There is no hint in Fish of a flexible operating system that executes as both a native operating system and a virtualized operating system under respective different conditions.

In view of the foregoing, it is clear that even if Silvester, Kishi, and Fish could be hypothetically combined, the hypothetical combination of the references would not have provided any hint of the claimed subject matter. Moreover, in view of the significant differences between the claimed subject matter and the teachings of Silvester, Kishi, and Fish, a person of ordinary skill in the art would not have been prompted to combine the teachings of the references to achieve the claimed subject matter. Specifically, while each of Silvester, Kishi, and Fish discloses the use of multiple **different** operating systems under different conditions, claim 1 recites that the flexible operating system executes as a native operating system and a virtualized operating system under different respective conditions.

In view of the foregoing, it is clear that the obviousness rejection of claim 1 and its dependent claims is erroneous.

The obviousness rejection of independent claim 27 and its dependent claims is similarly erroneous.

Reversal of the final rejection of the above claims is respectfully requested.

2. Claims 11-15.

Independent claim 1 recites a method comprising:

> implementing at least one operating system on a computer system;
>
> determining, by said computer system, whether said at least one operating system is a native operating system or a guest operating system on a virtual machine, wherein the determining is based on checking a variable set during a boot process of the computer system;

> said at least one operating system operating in a first manner if determined that it is a native operating system, wherein the native operating system operates in a non virtualized environment; and
>
> said at least one operating system operating in a second manner if determined that it is a guest operating system on a virtual machine, wherein the guest operating system operates in a virtual environment provided by the virtual machine.

Claim 11 is allowable over Silvester, Kishi, and Fish for reasons similar to those stated above with respect to claim 1. Specifically, the hypothetical combination of the references provide no teaching or hint of at least one operating system operating in a first manner if determined that it is a native operating system, and the at least one operating system operating in a second manner is determined that it is a guest operating system on a virtual machine, where the determination of whether the at least one operating system is a native operating system or a guest operating system is based on checking a variable set during a boot process of the computer system.

Also, for reasons similar to those stated with respect to claim 1, no reason existed that would have prompted a person of ordinary skill in the art to combine the teachings of Silvester, Kishi, and Fish to achieve the claimed subject matter.

The obviousness rejection of claim 11 and its dependent claims is therefore erroneous.

Reversal of the final rejection of the above claims is respectfully requested.

3. Claims 17, 18, 24, 25.

Independent claim 17 recites a computer system comprising, *inter alia*:

an operating system executable on at least one processor to:

> determine whether said operating system is running as a virtualized operating system or a native operating system, wherein the determining is based on checking a variable set during a boot process of the computer system; and
>
> adapt operation of said operating system depending on whether it is running as the virtualized operating system or native operating system, wherein the native operating system is configured to manage hardware resources in a non

> virtualized environment without the VMM, and wherein the virtualized operating system is configured to manage hardware resources using the VMM.

Claim 17 is allowable over Silvester, Kishi, and Fish for similar reasons as stated above with respect to claim 1. Specifically, the hypothetical combination of the references provide no teaching or hint of an operating system to determine (based on checking a variable set during a boot process) whether the operating system is running as a virtualized operating system or a native operating system, and to adapt operation of the operating system depending on whether it is running as the virtualized operating system or native operating system.

Moreover, for reasons similar to those stated above with respect to claim 1, no reason existed that would have prompted a person of ordinary skill in the art to combine the teachings of the references to achieve the claimed subject matter. Claim 17 and its dependent claims are therefore clearly non-obvious over Silvester, Kishi, and Fish.

Reversal of the final rejection of the above claims is respectfully requested.

4. Claims 35, 36, 40, 42.

Independent claim 35 recites a system comprising:

> at least one processor;
>
> a flexible operating system executable on the at least one processor and that is capable of acting as either a native operating system or as a virtualized operating system; and
>
> means for determining whether the flexible operating system is to be used as a native operating system in a non virtualized environment without a Virtual Machine Monitor (VMM) or as a virtualized operating system in a virtualized environment with the VMM, wherein the determining means checks a variable set during a boot process of the system for determining whether the flexible operating system is being used as the native or as the virtualized operating system.

Claim 35 is allowable over Silvester, Kishi, and Fish for reasons similar to those stated above with respect to claim 1. Specifically, the hypothetical combination of references fail to provide any hint of a flexible operating system that is capable of acting as either a native

operating system or as a virtualized operating system, in combination with means for determining whether the flexible operating system is to be used as a native operating system in a non-virtualized environment without a VMM or as a virtualized operating system in a virtualized environment with the VMM, where the determining means checks a variable set during a boot process for determining whether the flexible operating system is being used as the native or as the virtualized operating system.

Moreover, for reasons similar to those stated above with respect to claim 1, no reason existed that would have prompted a person of ordinary skill in the art to combine the teachings of the references to achieve the subject matter of claim 35.

Therefore, the obviousness rejection of claim 35 and its dependent claims is erroneous.

Reversal of the final rejection of the above claims is respectfully requested.

5. Claim 44.

Claim 44 depends from claim 1 and is therefore allowable for at least the same reasons as claim 1. Moreover, claim 44 further recites:

> wherein the at least one processor is configured to selectively set the variable to one of plural values during the boot process, a first of the plural values to indicate that the flexible operating system is to be used as the native operating system, and a second of the plural values to indicate that the flexible operating system is to be used as the virtualized operating system,
>
> wherein the flexible operating system is executable to check the variable during runtime after the boot process.

As purportedly disclosing the subject matter of claim 44, the Examiner cited Fish. 03/02/2010 Office Action at 10. Note that according to claim 44, the at least one processor is configured to selectively **set** the variable to one of plural values **during the boot process**, and the flexible operating system is executable to **check** the variable during **runtime after the boot process**. In Fish, Fig. 3 depicts a bootup phase in which a software extraction layer (SAL)

causes the bootstrap processor to load a master boot record such that the SAL can cause the bootstrap processor to determine whether the virtual mode operating system was selected. The checking of the master boot record **during the bootup phase** to determine whether the virtual mode operating system was selected, as taught by Fish, is different from the subject matter of claim 44, where the flexible operating system is executable to check the variable during **runtime after the boot process**. It appears that the Examiner is equating the "variable" of claim 44 with the master boot record of Fish—however, it is significant to note that the checking of the master boot record in Fish occurs during the bootup phase, not during runtime after the boot process as claimed.

Therefore, claim 44 is further allowable for the foregoing reasons.

Reversal of the final rejection of the above claim is respectfully requested.

6. Claim 45.

Claim 45 depends from claim 11 and is therefore allowable for at least the same reasons as claim 11. Moreover, claim 45 is also further allowable for similar reasons as stated above with respect to claim 44.

Reversal of the final rejection of the above claim is respectfully requested.

7. Claim 46.

Claim 46 depends from claim 17 and is therefore allowable for at least the same reasons as claim 17. Moreover, claim 46 is further allowable for similar reasons as stated above with respect to claim 44.

Reversal of the final rejection of the above claim is respectfully requested.

8. Claim 47.

Claim 47 depends indirectly from claim 35 and is therefore allowable for at least the same reasons as claim 35. Moreover, claim 47 is further allowable for similar reasons as claim 44.

Reversal of the final rejection of the above claim is respectfully requested.

B. Claims 3, 4, 8, 16, 19, 20, 22, 31, 33, 37, and 39 were rejected under 35 U.S.C. § 103(a) as unpatentable over Silvester in view of Kishi and in view Fish, and in view of Bennett (U.S. Patent Publication No. 2004/011732).

1. Claims 3, 4, 8, 16, 19, 20, 22, 31, 33, 37, 39.

In view of the allowability of base claims over Silvester, Kishi, and Fish, the obviousness rejection of the foregoing dependent claims over Silvester, Kishi, Fish, and Bennett has been overcome.

Reversal of the final rejection of the above claims is respectfully requested.

C. Claim 9 was rejected under 35 U.S.C. § 103(a) as unpatentable over Silvester in view of Kishi and in view of Fish, and in view of Bennett and further in view of Waldspurger (U.S. Patent No. 6,725,289).

1. Claim 9.

In view of the allowability of base claim 8 over Silvester, Kishi, Fish, and Bennett, the obviousness rejection of claim 9 over Silvester, Kishi, Fish, Bennett, and Waldspurger has been overcome.

Reversal of the final rejection of the above claim is respectfully requested.

CONCLUSION

In view of the foregoing, reversal of all final rejections and allowance of all pending claims is respectfully requested.

Respectfully submitted,

Date: July 21, 2010

/Dan C. Hu/

Dan C. Hu
Registration No. 40,025
TROP, PRUNER & HU, P.C.
1616 South Voss Road, Suite 750
Houston, TX 77057-2631
Telephone: (713) 468-8880
Facsimile: (713) 468-8883

VIII. APPENDIX OF APPEALED CLAIMS

Claims 2, 7, 10, 21, 23, 26, 28, 32, 34, 38, and 41 have been cancelled.

Claims 5 and 6 have been allowed.

The claims on appeal are:

1. A computer system comprising:

at least one processor; and

a flexible operating system executable on the at least one processor to:

determine whether said flexible operating system is being used as a native operating system or as a virtualized operating system on said computer system, wherein the determining is based on checking a variable set during a boot process of the computer system; and

execute in a first manner as a native operating system on the computer system in response to detecting that said flexible operating system is being used as the native operation system, and execute in a second manner as a virtualized operating system on said computer system in response to detecting that said flexible operating system is being used as the virtualized operating system,

wherein said flexible operating system is configured to operate in a non virtualized environment when said flexible operating system is being used as the native operating system, and is configured to operate in a virtualized environment when said flexible operating system is being used as the virtualized operating system.

3. The computer system of claim 1 wherein said flexible operating system executing in said second manner comprises said operating system acting as a paravirtualized operating system.

4. The computer system of claim 3 wherein said paravirtualized operating system is operable to make a call to a Virtual Machine Monitor (VMM) for performing at least one privileged operation.

8. The computer system of claim 1, wherein said flexible operating system is executable to further:

make a call to a Virtual Machine Monitor (VMM) for performing at least one privileged operation when the flexible operating system is executed in the second manner as the virtualized operating system.

9. The computer system of claim 8 wherein making the call to said VMM uses an Application Program Interface (API) defined for said VMM.

11. A method comprising:

implementing at least one operating system on a computer system;

determining, by said computer system, whether said at least one operating system is a native operating system or a guest operating system on a virtual machine, wherein the determining is based on checking a variable set during a boot process of the computer system;

said at least one operating system operating in a first manner if determined that it is a native operating system, wherein the native operating system operates in a non virtualized environment; and

said at least one operating system operating in a second manner if determined that it is a guest operating system on a virtual machine, wherein the guest operating system operates in a virtual environment provided by the virtual machine.

12. The method of claim 11 wherein said determining comprises:

said at least one operating system determining during runtime based on the variable whether the at least one operating system is being used as said native operating system or as said guest operating system on the virtual machine.

13. The method of claim 12 wherein said variable is a global variable.

14. The method of claim 11 wherein said first manner comprises said native operating system managing hardware resources of the computer system.

15. The method of claim 14 wherein said second manner comprises said guest operating system having access to the computer system hardware resources that are managed by a Virtual Machine Monitor (VMM).

16. The method of claim 15 wherein said guest operating system makes, for at least one privileged operation, a call to the VMM.

17. A computer system comprising:

at least one processor;

a virtual machine monitor (VMM); and

an operating system executable on the at least one processor to:

determine whether said operating system is running as a virtualized operating system or a native operating system, wherein the determining is based on checking a variable set during a boot process of the computer system; and

adapt operation of said operating system depending on whether it is running as the virtualized operating system or native operating system, wherein the native operating system is configured to manage hardware resources in a non virtualized environment without the VMM, and wherein the virtualized operating system is configured to manage hardware resources using the VMM.

18. The computer system of claim 17 wherein said variable is a global variable.

19. The computer system of claim 18 wherein said operating system is executable to check said value of said global variable before performing certain privileged operations.

20. The computer system of claim 17 wherein said operating system is executable to perform the determining by determining, before execution of certain privileged instructions, whether said operating system is running as the virtualized operating system or native operating system.

22. The computer system of claim 17 wherein said operating system when running as the virtualized operating system executes privileged instructions by making at least one call to the VMM.

24. The computer system of claim 17 wherein said operating system performs the determining by executing an instruction which, when the operating system is being used as the virtualized operating system, causes the VMM to set at least one configuration bit to a first value.

25. The computer system of claim 24 wherein said operating system performs the determining by further determining whether said operating system is running as the virtualized operating system or native operating system based at least in part on a determined value of at least one configuration bit after execution of said instruction.

27. A system comprising:

hardware resources;

a virtual machine monitor (VMM); and

a flexible operating system for managing said hardware resources, wherein said flexible operating system is operable to determine whether it is running in a virtualized environment or in a native, non virtualized environment, wherein the determining is based on checking a variable set during a boot process of the system, wherein said flexible operating system is operable to selectively execute in a first manner if determined that said flexible operating system is running in the native environment and in a second manner if determined that said flexible operating system is running in the virtualized environment, wherein in the first manner said flexible operating system is configured to manage said hardware resources without using the VMM, and wherein in the second manner said flexible operating system is configured to manage said hardware resources using the VMM.

29. The system of claim 27 wherein said first manner comprises acting as a native operating system.

30. The system of claim 27 wherein said second manner comprises acting as a virtualized operating system.

31. The system of claim 30 wherein said virtualized operating system is operable to make a call to the VMM for performing at least one privileged operation.

33. The system of claim 27 wherein said flexible operating system is configured to adapt its operation to make a call to said VMM for performance of at least one privileged instruction when said flexible operating system determines that said flexible operating system is running in the virtualized environment.

35. A system comprising:

at least one processor;

a flexible operating system executable on the at least one processor and that is capable of acting as either a native operating system or as a virtualized operating system; and

means for determining whether the flexible operating system is to be used as a native operating system in a non virtualized environment without a Virtual Machine Monitor (VMM) or as a virtualized operating system in a virtualized environment with the VMM, wherein the determining means checks a variable set during a boot process of the system for determining whether the flexible operating system is being used as the native or as the virtualized operating system.

36. The system of claim 35 wherein the determining means makes the determination during runtime of the system.

37. The system of claim 35 further comprising:

means for virtualizing resources of said system and multiplexing said resources among one or more virtualized operating systems.

39. The system of claim 35 wherein if determined that said flexible operating system is being used as the virtualized operating system, said flexible operating system is configured to act as the virtualized operating system.

40. The system of claim 35 wherein if determined that said flexible operating system is being used as the native operating system, said flexible operating system is configured to act in a first manner, and if determined that said flexible operating system is being used as the virtualized operating system, said flexible operating system is configured to act in a second manner.

42. The system of claim 35, wherein the virtualized operating system is configured to manage hardware resources of the system by using the VMM, and wherein the native operating system is configured to manage the hardware resources in the non virtualized environment without using the VMM.

43. The computer system of claim 1, wherein the virtualized operating system is configured to manage hardware resources of the system by using a virtual machine monitor (VMM), and wherein the native operating system is configured to manage the hardware resources in the non virtualized environment without using the VMM.

44. The computer system of claim 1, wherein the at least one processor is configured to selectively set the variable to one of plural values during the boot process, a first of the plural values to indicate that the flexible operating system is to be used as the native operating system, and a second of the plural values to indicate that the flexible operating system is to be used as the virtualized operating system,

wherein the flexible operating system is executable to check the variable during runtime after the boot process.

45. The method of claim 11, further comprising:

selectively setting the variable to one of plural values during the boot process, a first of the plural values to indicate that the at least one operating system is to be used as the native operating system, and a second of the plural values to indicate that the at least one operating system is to be used as the guest operating system; and

the at least one operating system checking the variable during runtime after the boot process.

46. The computer system of claim 17, wherein the at least one processor is configured to selectively set the variable to one of plural values during the boot process, a first of the plural values to indicate that said operating system is to be used as the native operating system, and a second of the plural values to indicate that said operating system is to be used as the virtualized operating system, wherein said operating system is executable to check the variable during runtime after the boot process.

47. The system of claim 37, further comprising at least one processor configured to selectively set the variable to one of plural values during the boot process, a first of the plural values to indicate that the flexible operating system is to be run in the native environment, and a second of the plural values to indicate that the flexible operating system is to be fun in the virtualized environment,

wherein the flexible operating system is operable to check the variable during runtime after the boot process.

IX. EVIDENCE APPENDIX

None.

X. RELATED PROCEEDINGS APPENDIX

None.